Safely composing security protocols

Véronique Cortier, Jérémie Delaitre, Stéphanie Delaune

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Safely composing security protocols

Véronique Cortier — Jérémie Delaitre — Stéphanie Delaune

N° 6234

Juin 2007

Thème SYM

ISSN 0249-6399 ISRN INRIA/RR--6234--FR+ENG

Safely composing security protocols

Véronique Cortier* , Jérémie Delaitre * , Stéphanie Delaune*

Thème SYM — Systèmes symboliques
Projets Cassis

Rapport de recherche n° 6234 — Juin 2007 — 26 pages

Abstract: Security protocols are small programs that are executed in hostile environments. Many results and tools have been developed to formally analyze the security of a protocol in the presence of active attackers that may block, intercept and send new messages. However even when a protocol has been proved secure, there is absolutely no guarantee if the protocol is executed in an environment where other protocols, possibly sharing some common identities and keys like public keys or long-term symmetric keys, are executed.

In this paper, we show that security of protocols can be easily composed. More precisely, we show that whenever a protocol is secure, it remains secure even in an environment where arbitrary protocols are executed, provided each encryption contains some tag identifying each protocol, like *e.g.* the name of the protocol.

Key-words: verification, security protocols, composition

* LORIA, CNRS & INRIA project Cassis, Nancy, France. This work has been partly supported by the RNTL project POSÉ and the ARA SSIA Formacrypt

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,
615, rue du Jardin Botanique, BP 101, 54602 Villers-Lès-Nancy (France)
Téléphone : +33 3 83 59 30 00 — Télécopie : +33 3 83 27 83 19

Composition de protocoles de sécurité

Résumé : Les protocoles de sécurité sont des petits programmes qui s'exécutent dans un environnement hostile. De nombreux résultats permettent de réaliser une analyse formelle de ces protocoles en présence d'un intrus actif qui peut bloquer, intercepter et envoyer de nouveaux messages. Cependant ces résultats ne fournissent aucune garantie de sécurité dans le cas où le protocole est exécuté dans un environnement où d'autres protocoles, partageant éventuellement des clefs, interviennent.

Dans ce papier, nous montrons que la sécurité des protocoles se composent. Plus précisément, nous montrons que lorsqu'un protocole est sûr, il le reste même lorsque d'autres protocoles sont exécutés, pourvu que chaque message chiffré du protocole contienne un identifiant, comme par exemple le nom du protocole.

Mots-clés : vérification, protocoles de sécurité, composition

1 Introduction

Security protocols are small programs that aim at securing communications over a public network like the Internet. Considering the increasing size of networks and their dependence on cryptographic protocols, a high level of assurance is needed in the correctness of such protocols. The design of such protocols is difficult and error-prone; many attacks have been discovered even several years after the publication of a protocol. Consequently, there has been a growing interest in applying formal methods for validating cryptographic protocols and many results have been obtained. The main advantage of the formal approach is its relative simplicity which makes it amenable to automated analysis. For example, the secrecy preservation is co-NP-complete for a bounded number of sessions [17], and decidable for an unbounded number of sessions under some additional restrictions (*e.g.* [2, 4, 18]). Many tools have also been developed to automatically verify cryptographic protocols like [14, 3].

However even when a protocol has been proved secure for an unbounded number of sessions, against a fully active adversary that can intercept, block and send new messages, there is absolutely no guarantee if the protocol is executed in an environment where other protocols, possibly sharing some common identities and keys like public keys or long-term symmetric keys, are executed. This is however very likely to happen since a user connected to the Internet for example, usually uses simultaneously several protocols with the same identity. The interaction with the other protocols may dramatically damage the security of a protocol. Consider for example the two following naive protocols.

$$P_1: \quad A \to B: \quad \{s\}_{\mathrm{pub}(B)} \qquad\qquad P_2: \quad \begin{array}{ll} A \to B: & \{N_a\}_{\mathrm{pub}(B)} \\ B \to A: & N_a \end{array}$$

In protocol P_1, the agent A simply sends a secret s encrypted under B's public key. In protocol P_2, the agent sends some fresh nonce to B encrypted under B's public key. The agent B acknowledges A's message by forwarding A's nonce. While P_1 executed alone easily guarantees the secrecy of s, even against an active adversary, the secrecy of s is no more guaranteed when the protocol P_2 is executed. Indeed, an adversary may use the protocol P_2 as an oracle to decrypt any message. More realistic examples illustrating interactions between protocols can be found in *e.g.* [13].

The purpose of this paper is to investigate sufficient and rather tight conditions for a protocol to be safely used in an environment where other protocols may be executed as well. Our main contribution is to show that whenever a protocol is proved secure when it is executed alone, its security is not compromised by the interactions with any other protocol, provided each protocol is given an identifier (*e.g.* the protocol's name) that should appear in any encrypted message. Continuing our example, let us consider the two slightly modified protocols.

$$P_1': \quad A \to B: \quad \{1, s\}_{\mathrm{pub}(B)} \qquad\qquad P_2': \quad \begin{array}{ll} A \to B: & \{2, N_a\}_{\mathrm{pub}(B)} \\ B \to A: & N_a \end{array}$$

Applying our result, we immediately deduce that P_1' can be safely executed together with P_2', without compromising the secrecy of s.

The idea of adding an identifier in encrypted messages is not novel. This rule is in the same spirit as those proposed in the paper of Abadi and Needham on prudent engineering practice for cryptographic protocols [1] (principle 10). The use of unique protocol identifiers is also recommended in [13, 5] and has also been used in the design of fail-stop protocols [11]. However, to the best of our knowledge, it has never been proved that it is sufficient for securely executing several protocols in the same environment. Note that some other results also use tags for different purposes. For instance, Blanchet uses tags to exhibit a decidable class [4] but his tagging policy is stronger since any two encrypted subterm in a protocol have to contain different tags.

The result the most closely related to ours is the one of Guttman and Thayer [12]. They show that two protocols can be safely executed together without damaging interactions, as soon as the protocols are "independent". The independence hypothesis requires in particular that the set of encrypted messages that the two protocols handle should be different. As in our case, this can be ensured by giving each protocol a distinguishing value that should be included in the set of encrypted messages that the protocol handles. However, the major difference with our result is that this hypothesis has to hold on any valid *execution* of the protocol. In particular, considering again the protocol P_2', an agent should not accept a message of the form $\{2, \{1, m\}_k\}_{\mathrm{pub}(B)}$ while he might not be able to decrypt the inside encryption and detect that it contains the wrong identifier. In particular, their result do not allow to conclude when no typing hypothesis is assumed (that is, when agents are not required to check the type of each component of a message) or for protocols with cyphertext forwarding, that is, when agents have to forward unknown message components.

Datta *et al.* [9, 10] have also studied secure protocol composition in a more broader sense: protocols can be composed in parallel, sequentially or protocols may use other protocols as components. However, they do not provide any syntactic conditions for a protocol P to be safely executed in parallel with other protocols. For any protocol P' that might be executed in parallel, they have to prove that the two protocols P and P' satisfy each other invariants. Their approach is thus rather designed for component-based design of protocols.

2 Models for security protocols

2.1 Syntax

Cryptographic primitives are represented by *function symbols*. More specifically, we consider the *signature* $\mathcal{F} = \{\mathrm{enc}, \mathrm{enca}, \mathrm{sign}, \langle\,\rangle, \mathrm{pub}, \mathrm{priv}\}$ together with arities of the form $\mathrm{ar}(f) = 2$ for the four first symbols and $\mathrm{ar}(f) = 1$ for the two last ones. The symbol $\langle\,\rangle$ represents the pairing function. The terms $\mathrm{enc}(m, k)$ and $\mathrm{enca}(m, k)$ represent respectively the message m encrypted with the symmetric (resp. asymmetric) key k. The term $\mathrm{sign}(m, k)$ represents the message m signed by the key k. The terms $\mathrm{pub}(a)$ and $\mathrm{priv}(a)$ represent respectively the public and private keys of an agent a. We fix an infinite set of *names* $\mathcal{N} = \{a, b \ldots\}$ among which we distinghuish two particular names init and stop; and an infinite set of *variables*

$\mathcal{X} = \{x, y \ldots\}$. The set of Terms is defined inductively by

t	::=		term
	$\mid$	x	variable x
	$\mid$	a	name a
	$\mid$	$f(a)$	application of symbol $f \in \{\text{pub}, \text{priv}\}$ on a name
	$\mid$	$f(t_1, t_2)$	application of symbol $f \in \{\text{enc}, \text{enca}, \text{sign}, \langle\,\rangle\}$

As usual, we write $vars(t)$ (resp. names(t)) for the set of variables (resp. names) occurring in t. A term is *ground* or *closed* if and only if it has no variables. We write $St(t)$ for the set of *subterms* of a term t. This notion is extended as expected to sets of terms. *Extended names* are names or terms of the form pub(a), priv(a). The set of *extended names of a term* t, denoted by n(t), is n$(t) =$ names$(t) \cup \{\text{pub}(t), \text{priv}(t) \mid \text{pub}(t)$ or priv$(t) \in St(t)\}$. For example, we have that n$(\text{enc}(a, \text{pub}(b))) = \{a, b, \text{pub}(b), \text{priv}(b)\}$. Substitutions are written $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ with $\text{dom}(\sigma) = \{x_1, \ldots, x_n\}$. The substitution σ is *closed* if and only if all the t_i are closed. The application of a substitution σ to a term t is written $\sigma(t)$ or $t\sigma$.

2.2 Intruder capabilities

The ability of the intruder is modelled by a deduction system described in Figure 1 and corresponds to the usual Dolev-Yao rules. The first line describes the *composition* rules. We call these deduction rules pairing, symmetric and asymmetric encryption and signature respectively. The two last lines describe the *decomposition* rules and the axiom. We call these rules first and second projection, symmetric and asymmetric decryption respectively. Intuitively, these deduction rules say that an intruder can compose messages by pairing, encrypting and signing messages provided he has the corresponding keys. Conversely, it can decompose messages by projecting or decrypting provided it has the decryption keys. For signatures, the intruder is also able to *verify* whether a signature sign(m, k) and a message m match (provided she has the verification key), but this does not give her any new message. That is why this capability is not represented in the deduction system. We also consider an optional rule

$$\frac{T \vdash \text{sign}(u, \text{priv}(v))}{T \vdash u}$$

that expresses that an intruder can retrieve the whole message from its signature. This property may or may not hold depending on the signature scheme, and that is why this rule is optional. Our results hold in both cases (that is, when the deduction relation $\vdash$ is defined with or without this rule).

A term u is *deducible* from a set of terms T, denoted by $T \vdash u$ if there exists a *proof i.e.* a tree such that the root is $T \vdash u$, the leaves are of the form $T \vdash v$ with $v \in T$ (*axiom* rule) and every intermediate node is an instance of one of the rules of the deduction system.

$$\frac{T \vdash u \quad T \vdash v}{T \vdash \langle u, v\rangle} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{enc}(u, v)} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{enca}(u, v)} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{sign}(u, v)}$$

$$\frac{T \vdash \langle u, v\rangle}{T \vdash u} \qquad \frac{T \vdash \langle u, v\rangle}{T \vdash v} \qquad \frac{T \vdash \mathrm{enc}(u, v) \quad T \vdash v}{T \vdash u}$$

$$\frac{T \vdash \mathrm{enca}(u, \mathrm{pub}(v)) \quad T \vdash \mathrm{priv}(v)}{T \vdash u} \qquad \frac{T \vdash \mathrm{sign}(u, \mathrm{priv}(v))}{T \vdash u}\ \textit{(optional)} \qquad \frac{}{T \vdash u}\ u \in T$$

Figure 1: Intruder deduction system.

Example 1 *The term $\langle k_1, k_2\rangle$ is deducible from the set $T_1 = \{\mathrm{enc}(k_1, k_2), k_2\}$. A proof of $T_1 \vdash \langle k_1, k_2\rangle$ is:*

$$\dfrac{\dfrac{T_1 \vdash \mathrm{enc}(k_1, k_2) \quad T_1 \vdash k_2}{T_1 \vdash k_1} \qquad T_1 \vdash k_2}{T_1 \vdash \langle k_1, k_2\rangle}$$

2.3 Protocols

We consider protocols specified in a language similar to the one of [17] allowing parties to exchange messages built from identities and randomly generated nonces using public key, symmetric encryption and digital signatures. The individual behavior of each protocol participant is defined by a *role* describing a sequence of message receptions/transmissions, and a k-party protocol is given by k such roles.

Definition 1 (Roles and protocols) *The set* Roles *of roles for protocol participants is the set of sequences of the form* $(\mathsf{rcv}_1, N_1, \mathsf{snd}_1) \cdots (\mathsf{rcv}_\ell, N_\ell, \mathsf{snd}_\ell)$ *where each element, called* rule*, satisfies* $(\mathsf{rcv}_i, N_i, \mathsf{snd}_i) \in \mathsf{Terms} \times 2^{\mathcal{X}} \times \mathsf{Terms}$*, and for any variable, $x \in vars(\mathsf{snd}_i)$ implies $x \in \bigcup_{1 \le j \le i} N_j \cup vars(\mathsf{rcv}_j)$.*

The length *of a role is the number of elements in its sequence. A k-*party protocol *is a mapping $\Pi : [k] \to \mathsf{Roles}$, where $[k] = \{1, 2, \ldots, k\}$.*

The last condition ensures that each variable which appears in a sent term is either a nonce or has been introduced in a previously received message. The set of variables, names or extended names of a protocol is defined as expected, considering all the terms occurring in the role's specification.

The j^{th} role of a protocol Π is denoted by $(\mathsf{rcv}_1^j \xrightarrow{N_1^j} \mathsf{snd}_1^j) \cdots (\mathsf{rcv}_{k_j}^j \xrightarrow{N_{k_j}^j} \mathsf{snd}_{k_j}^j)$. It specifies the messages to be sent/received by the party executing the role: at step i, the j^{th} party expects to receive a message conformed to rcv_i^j, instantiate the variables of N_i^j with fresh

names and returns the message snd_i^j. We assume the sets N_i^j to be pairwise disjoint. The special constants init and stop will be used to specify that no message is expected or sent.

The *composition* of two protocols Π_1 and Π_2, denoted by $\Pi_1 \mid \Pi_2$ is the protocol obtained by the union of the roles of Π_1 and Π_2. If $\Pi_1 : [k_1] \to \mathsf{Roles}$ and $\Pi_2 : [k_2] \to \mathsf{Roles}$, then $\Pi = \Pi_1 \mid \Pi_2 : [k_1 + k_2] \to \mathsf{Roles}$ with $\Pi(i) = \Pi_1(i)$ for any $1 \leq i \leq k_1$ and $\Pi(k_1 + i) = \Pi_2(i)$ for any $1 \leq i \leq k_2$.

Example 2 *Consider the famous Needham-Schroeder asymmetric key authentication protocol [16] designed for mutual authentication.*

$$\begin{aligned} A \to B : &\quad \{N_a, A\}_{\mathrm{pub}(B)} \\ B \to A : &\quad \{N_a, N_b\}_{\mathrm{pub}(A)} \\ A \to B : &\quad \{N_b\}_{\mathrm{pub}(B)} \end{aligned}$$

The agent A sends to B his name and a fresh nonce (a randomly generated value) encrypted with the public key of B. The agent B answers by copying A's nonce and adds a fresh nonce N_B, encrypted by A's public key. The agent A acknowledges by forwarding B's nonce encrypted by B's public key. For instance, let a, b, and c be three agent names. The role $\Pi(1)$ *corresponding to the first participant played by a talking to c is:*

$$(\mathsf{init} \overset{\{X\}}{\to} \mathrm{enca}(\langle X, a\rangle, \mathrm{pub}(c))), (\mathrm{enca}(\langle X, x\rangle, \mathrm{pub}(a)) \overset{\emptyset}{\to} \mathrm{enca}(x, \mathrm{pub}(c)))$$

The role $\Pi(2)$ *corresponding to the second participant played by b with a is:*

$$(\mathrm{enca}(\langle y, a\rangle, \mathrm{pub}(b)) \overset{\{Y\}}{\to} \mathrm{enca}(\langle y, Y\rangle, \mathrm{pub}(a))), (\mathrm{enca}(Y, \mathrm{pub}(b)) \overset{\emptyset}{\to} \mathsf{stop})$$

Note that there is also a role corresponding to the first participant played by a talking to b for example. If more agent identities need to be considered, then the corresponding roles should be added to the protocol. It has been shown however that two agents are sufficient (one honest and one dishonest) for proving security properties [6].

Clearly, not all protocols written using the syntax above are meaningful. In particular, some of them might not be *executable*. A precise definition of executability is not relevant for our result. We use instead a weaker hypothesis (see Section 3). In particular, our combination result also holds for non executable protocols that satisfy our hypothesis.

2.4 Constraint systems

Constraint systems are quite common (see e.g. [17, 7, 8]) in modeling security protocols. They are used to specify secrecy preservation of security protocols under a particular, finite scenario. We recall here their formalism and we show in the next section that the secrecy preservation problem for an *unbounded number of sessions* can be specified using (infinite) families of constraint systems.

Definition 2 (constraint system) *A* constraint system $\mathcal{C}$ *is either* $\bot$ *or a finite set of expressions* $T \Vdash u$*, called* constraints*, where* T *is a non empty set of terms, called the* left-hand side *of the constraint and* u *is a term, called the* right-hand side *of the constraint, such that:*

- *the left-hand sides of all constraints are totally order by inclusion;*
- *if* $x \in vars(T)$ *for some* $(T \Vdash u) \in \mathcal{C}$ *then*
$$T_x \stackrel{\mathsf{def}}{=} \min\{T' \mid (T' \Vdash u') \in \mathcal{C} \text{ and } x \in vars(u')\}$$
exists and $T_x \subsetneq T$*.*

A solution *of* $\mathcal{C}$ *is a closed substitution* θ *such that for every* $(T \Vdash u) \in \mathcal{C}$*, we have that* $T\theta \vdash u\theta$*. The empty constraint system is always satisfiable whereas* $\bot$ *denotes an unsatisfiable system.*

A constraint system $\mathcal{C}$ is usually denoted as a conjunction of constraints $\mathcal{C} = \bigwedge_{1 \le i \le n}(T_i \Vdash u_i)$ with $T_i \subseteq T_{i+1}$, for all $1 \le i < n$. The second condition in Definition 2 says that if $x \in vars(T_i)$ then $\exists j < i$ such that $T_j = T_x$ and $T_j \subsetneq T_i$. In other words, each time a variable occurs first in some right-hand side, it must not have occurred before in some left-hand side. The left-hand side of a constraint system usually represents the messages sent on the network.

2.5 Secrecy

We define the general secrecy preservation problem for an unbounded number of sessions, using infinite families of constraint systems. A role may be executed in several sessions, using different nonces at each session. Moreover, since the adversary may block, redirect and send new messages, all the sessions might be interleaved in many ways. This is captured by the notion of *scenario*.

Definition 3 (scenario) *A* scenario for a protocol $\Pi : [k] \to \mathsf{Roles}$ *is a sequence of the form* $(r_1, s_1) \cdots (r_n, s_n)$ *such that* $1 \le r_i \le k$*,* $s_i \in \mathbb{N}$*, the number of identical occurrences of a pair* (r, s) *is smaller than the length of the role* r*, and whenever* $s_i = s_j$ *then* $r_i = r_j$*.*

The numbers r_i and s_i represent respectively the involved role and the session number. The last condition ensures that a session number is not reused on other roles. Given a scenario sc, we say that $(r, s) \in \mathsf{sc}$ if (r, s) is an element of the sequence sc. Let $\Pi = \Pi_1 \mid \Pi_2$ be a protocol obtained by composition of Π_1 and Π_2 and let sc be a scenario for Π. The scenario $\mathsf{sc}|_{\Pi_1}$ is simply the sequence obtained from sc by removing any element (r, s) where r is a role of Π_2. Given a scenario, we can define a sequence of rules that corresponds to the sequence of expected and sent messages.

Definition 4 *Given a scenario* $\mathsf{sc} = (r_1, s_1) \cdots (r_n, s_n)$ *for a* k*-party protocol* Π*, the* sequence of rules $(u_1, v_1) \cdots (u_n, v_n)$ associated to sc *is defined as follows.*

$$\text{Let } \Pi(j) = (\mathsf{rcv}_1^j \stackrel{N_1^j}{\to} \mathsf{snd}_1^j) \cdots (\mathsf{rcv}_{k_j}^j \stackrel{N_{k_j}^j}{\to} \mathsf{snd}_{k_j}^j) \text{ for } 1 \le j \le k.$$

Let $p_i = \#\{(r_j, s_j) \in \mathsf{sc} \mid j \leq i, r_j = r_i\}$, i.e. *the number of previous occurrences in* sc *of the role* r_i. *We have* $p_i \leq k_{r_i}$ *and* $(u_i, v_i) = (\mathsf{rcv}^{r_i}_{p_i}\sigma_{r_i,s_i}, \mathsf{snd}^{r_i}_{p_i}\sigma_{r_i,s_i})$, *where*

- $\mathrm{dom}(\sigma_{r,s}) = \bigcup_{1\leq i\leq k_r} (N^r_i \cup vars(\mathsf{rcv}^r_i))$, i.e. *variables occurring in* $\Pi(r)$,
- $\sigma_{r,s}(x) = n_{x,s}$ *if* $x \in \bigcup_{1\leq i\leq k_r} N^r_i$, *where* $n_{x,s}$ *is a name.*
- $\sigma_{r,s}(x) = x_s$ *otherwise, where* x_s *is a variable.*

We assume that names (resp. variables) with different indexes are pairwise different and also different from the names (resp. variables) occurring in Π,

We say that a protocol preserves the secrecy of a data if it preserves its secrecy for any scenario. In particular, the secrecy of the data must be preserved for any possible instances of its fresh values (*e.g.* nonces and keys).

Definition 5 (secrecy) *A protocol* Π preserves the secrecy *of a term* m for the initial knowledge T_0 *if for any scenario* sc *for* Π, *for any role number* $1 \leq i \leq k$, *for any session number* $s_i \in \mathbb{N}$ *that either corresponds to role* i, *that is* $(i, s_i) \in \mathsf{sc}$ *or does not appear in the scenario, that is* $\forall j, (j, s_i) \notin \mathsf{sc}$, *the following constraint system is not satisfiable*

$$T'_0 \Vdash u_1 \wedge \bigwedge_{1\leq i<n} (T'_0 \cup \{v_1, \ldots, v_i\} \Vdash u_{i+1}) \wedge (T'_0 \cup \{v_1, \ldots, v_n\} \Vdash m\sigma_{1,s_1} \cdots \sigma_{k,s_k})$$

where $T'_0 = T_0 \cup \{\mathsf{init}\}$ *and* $(u_1, v_1) \cdots (u_n, v_n)$ *is the sequence of rules associated to* sc *and* $\sigma_{r,s}$ *is the substitution defined in Definition 4.*

The initial knowledge typically contains the names and the public keys of all agents and the private keys of all dishonest agents.

Example 3 *Consider again the Needham-Schroeder protocol. Let* $\Pi(1)$ *and* $\Pi(2)$ *the two roles introduced in Example 2. This protocol is well-known to be insecure w.r.t.* $m = Y$ *and* $T_0 = \{\mathrm{priv}(c), \mathrm{pub}(c), a, b, \mathrm{pub}(a), \mathrm{pub}(b)\}$. *Let* s_1 *and* s_2 *be two session numbers* ($s_1 \neq s_2$) *and consider* $\mathsf{sc} = (1, s_1)\,(2, s_2)\,(1, s_1)\,(2, s_2)$. *The constraint system* $\mathcal{C}$ *associated to* T_0, sc *and* $m\sigma_{1,s_1}\sigma_{2,s_2} = n_{Y,s_2}$ *(according to Definition 5) is given below.*

$$\mathcal{C} := \left\{ \begin{array}{rcl} T_0, \mathsf{init} & \Vdash & \mathsf{init} \\ T_1 \stackrel{\mathsf{def}}{=} T_0, \mathsf{init}, \mathrm{enca}(\langle n_{X,s_1}, a\rangle, \mathrm{pub}(c)) & \Vdash & \mathrm{enca}(\langle y_{s_2}, a\rangle, \mathrm{pub}(b)) \\ T_2 \stackrel{\mathsf{def}}{=} T_1, \mathrm{enca}(\langle y_{s_2}, n_{Y,s_2}\rangle, \mathrm{pub}(a)) & \Vdash & \mathrm{enca}(\langle n_{X,s_1}, x_{s_1}\rangle, \mathrm{pub}(a)) \\ T_2, \mathrm{enca}(x_{s_1}, \mathrm{pub}(c)) & \Vdash & \mathrm{enca}(n_{Y,s_2}, \mathrm{pub}(b)) \\ T_2, \mathrm{enca}(x_{s_1}, \mathrm{pub}(c)) & \Vdash & n_{Y,s_2} \end{array} \right.$$

The substitution $\sigma = \{y_{s_2} \mapsto n_{X,s_1}, x_{s_1} \mapsto n_{Y,s_2}\}$ *is a solution of* $\mathcal{C}$.

3 Composition result

3.1 Hypothesis

Even if a protocol is secure for an unbounded number of sessions, its security may collapse if the protocol is executed in an environment where other protocols sharing some common keys are executed. We have seen a first example in introduction. To avoid a cyphertext from a protocol Π_1 to be decrypted in an another protocol Π_2, we introduce the notion of *well-tagged* protocol.

Definition 6 (well-tag, α-tag) *Let α be a term. We say that a term t is α-tagged if for every $t' \in St(t)$ of the form $t' = \mathrm{enc}(t_1, t_2)$, $t' = \mathrm{enca}(t_1, t_2)$, or $t' = \mathrm{sign}(t_1, t_2)$, we have $t_1 = \langle \alpha, t_1' \rangle$ for some term t_1'. A term is said* well-tagged *if it is α-tagged for some term α.*

A protocol Π is α-tagged is any term occurring in the role of the protocol is α-tagged. A protocol is said well-tagged *if it is α-tagged for some term α.*

Requiring that a protocol is well-tagged can be very easily achieved in practice: it is sufficient for example to add the name of the protocol in each encrypted term. Moreover, note that (as opposite to [12]) this does not require that the agents check that nested encrypted terms are correctly tagged. For example, let Π be a protocol with one role as follows:

$$\Pi(1) = (\mathrm{enca}(\langle \alpha, x \rangle, \mathrm{pub}(a)) \rightarrow \mathrm{enca}(\langle \alpha, x \rangle, \mathrm{pub}(b))).$$

The protocol Π is α-tagged and still the message $\mathrm{enca}(\langle \alpha, \mathrm{enc}(a, k) \rangle, \mathrm{pub}(a))$ (which is not α-tagged) would be accepted by the agent playing the role.

Tagging protocols is not sufficient, indeed critical long-term keys should not be revealed in clear. Consider for example the following two well-tagged protocols

$$P_3 : \quad A \rightarrow B : \quad \{\alpha, s\}_{k_{ab}} \qquad\qquad P_4 : \quad A \rightarrow B : \quad k_{ab}$$

The security of protocol P_3 is again compromised by the execution of P_4. Thus we will require that long-term keys (except possibly the public ones) do not occur in plaintext in the protocol.

Definition 7 (plaintext) *The set $plaintext(t)$ of* plaintext *of a term t is the set of extended names and variables, that is recursively defined as follows.*

$plaintext(u) = \{u\}$	*if u is a variable or a name*
$plaintext(f(u)) = \{f(u)\}$	*for $f \in \{\mathrm{pub}, \mathrm{priv}\}$*
$plaintext(\langle u_1, u_2 \rangle) = plaintext(u_1) \cup plaintext(u_2)$	
$plaintext(f(u_1, u_2)) = plaintext(u_1)$	*for $f \in \{\mathrm{enc}, \mathrm{enca}, \mathrm{sign}\}$*

This notation is extended to set of terms and protocols as expected .

Some weird protocols may still reveal critical keys in a hidden way. Consider for example the following one role (α-tagged) protocol.

$$\Pi(1) = (\mathsf{init} \rightarrow \mathrm{enc}(\langle \alpha, a \rangle, k_{ab})),\ (\mathrm{enc}(\langle \alpha, a \rangle, x) \rightarrow x)$$

While the long-term key k_{ab} does not appear in plaintext, the key k_{ab} is revealed after simply one normal execution of the role. This protocol is however not realistic since an unknown value cannot be learned (and sent) if it does not appear previously in plaintext. Thus we will further require (Condition 2 of Theorem 1) that a variable occurring in plaintext in a sent message, has to previously occur in plaintext in a received message.

3.2 Composition theorem

We show that any two well-tagged protocols can be safely composed as soon as they use different tags and that critical long-term keys do not appear in plaintext.

Theorem 1 *Let Π_1 and Π_2 be two well-tagged protocols such that Π_1 is α-tagged and Π_2 is β-tagged with $\alpha \neq \beta$. Let T_0 (intuitively the initial knowledge of the intruder) be a set of extended names. Let $\mathsf{KC} = (\mathrm{n}(\Pi_1) \cup \mathrm{n}(\Pi_2)) \setminus T_0$ be the set of* critical extended names. *Let m be a term constructed from Π_1 such that m is α-tagged and $vars(m) \subseteq vars(\Pi_1)$. Moreover, we assume that*

1. *critical extended names do not appear in plaintext, that is*
$$\mathsf{KC} \cap (plaintext(\Pi_1) \cup plaintext(\Pi_2)) = \emptyset.$$
2. *for any role $(\mathsf{rcv}_1 \stackrel{N_1}{\rightarrow} \mathsf{snd}_1) \cdots (\mathsf{rcv}_k \stackrel{N_k}{\rightarrow} \mathsf{snd}_k)$ of Π_1 or Π_2, for any variable $x \in plaintext(\mathsf{snd}_i)$, we have $x \in \bigcup_{1 \leq j \leq i} N_j \cup \{plaintext(\mathsf{rcv}_j)\}$.*

Then Π_1 preserves the secrecy of m for the initial knowledge T_0 if and only if $\Pi_1 \mid \Pi_2$ preserves the secrecy of m for T_0.

We have seen in Section 3.1 that conditions 1 and 2 are necessary conditions. Moreover, condition 2 will be satisfied by any realistic (executable) protocol. We require that terms from Π_1 and Π_2 are tagged with distinct tags for simplicity. The key condition is actually that for any encrypted (or signed) subterm t_1 of Π_1 and for any encrypted (or signed) subterm t_2 of Π_2, the terms t_1 and t_2 cannot be unified.

Theorem 1 is proved by contradiction. Assume that $\Pi_1 \mid \Pi_2$ does not preserve the secrecy of m for T_0. It means that there exists a scenario sc for $\Pi_1 \mid \Pi_2$ such that the constraint system associated to sc, T_0 and m is satisfiable. Proposition 1 ensures that in this case, there exists a scenario sc' for Π_1 such that the constraint system associated to sc', T_0 and m is satisfiable, which means that Π_1 does not preserve the secrecy of m for some initial knowledge T_0, contradiction.

Proposition 1 *Let $\Pi_1 = [k_1] \to \mathsf{Roles}$, $\Pi_2 = [k_2] \to \mathsf{Roles}$, T_0 and m defined as in Theorem 1 and satisfying the conditions* 1 *and* 2. *Let $k = k_1 + k_2$ and* sc *be a scenario for $\Pi_1 \mid \Pi_2$. For any role number $1 \leq i \leq k$, let $s_i \in \mathbb{N}$ such that $(i, s_i) \in \mathsf{sc}$ or $\forall j, (j, s_i) \notin \mathsf{sc}$. Let $\mathcal{C}$ be the constraint system associated to* sc, *T_0 and $m\sigma_{1,s_1} \cdots \sigma_{k,s_k}$. Let $\mathsf{sc}' = \mathsf{sc}|_{\Pi_1}$ and $\mathcal{C}'$ be the constraint system associated to* sc', *T_0 and $m\sigma_{1,s_1} \cdots \sigma_{k_1,s_{k_1}}$. If $\mathcal{C}$ is satisfiable, then $\mathcal{C}'$ is also satisfiable.*

The next section is devoted to the (sketch of) proof of this proposition.

4 Proof of our combination result

To prove our decision procedure, we first refine an existing decision procedure for solving constraint systems. Several decision procedures already exist [15, 7, 8, 17] for solving constraint systems. Some of them [15, 7, 8] are based on a set of simplification rules allowing a general constraint system to be reduced to some simpler one, called *solved*, on which satisfiability can be easily decided. A constraint system is said *solved* [8] if it is different from $\bot$ and if each of its constraints is of the form $T \Vdash x$, where x is a variable. Note that the empty constraint system is solved. Solved constraint systems are particularly simple since they always have a solution. Indeed, let T_1 be the smallest (w.r.t. inclusion) left hand side of a constraint. From the definition of a constraint system we have that T_1 is non empty and has no variable. Let $t \in T_1$. Then the substitution τ defined by $x\tau = t$ for every variable x is a solution since $T \vdash x\theta$ for any constraint $T \Vdash x$ of the solved constraint system.

The *simplification rules* we consider are given below. All the rules are indexed by a substitution (when there is no index then the identity substitution is implicitly considered). We write $\mathcal{C} \rightsquigarrow_\sigma^* \mathcal{C}'$ if there are $\mathcal{C}_1, \ldots, \mathcal{C}_n$ such that $\mathcal{C} \rightsquigarrow_{\sigma_0} \mathcal{C}_1 \rightsquigarrow_{\sigma_1} \ldots \rightsquigarrow_{\sigma_n} \mathcal{C}'$ and $\sigma = \sigma_0\sigma_1 \ldots \sigma_n$. Our rules are the same than in [8] except that we forbid unification of terms headed by $\langle\, \rangle$. We show that it still forms a complete decision procedure (see Appendix A). Correction and termination are still ensured by [8].

R_1 :	$\mathcal{C} \wedge T \Vdash u$	$\rightsquigarrow$	$\mathcal{C}$	if $T \cup \{x \mid T' \Vdash x \in \mathcal{C}, T' \subsetneq R\} \vdash u$
R_2 :	$\mathcal{C} \wedge T \Vdash u$	$\rightsquigarrow_\sigma$	$\mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma$	if $\sigma = \mathrm{mgu}(t, u)$ where $t \in St(T)$, $t \neq u$, and t, u are neither variables nor pairs
R_3 :	$\mathcal{C} \wedge T \Vdash u$	$\rightsquigarrow_\sigma$	$\mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma$	if $\sigma = \mathrm{mgu}(t_1, t_2)$, $t_1, t_2 \in St(T)$, $t_1 \neq t_2$, and t_1, t_2 are neither variables nor pairs
R_4 :	$\mathcal{C} \wedge T \Vdash u$	$\rightsquigarrow$	$\bot$	if $vars(T, u) = \emptyset$ and $T \not\vdash u$
R_5 :	$\mathcal{C} \wedge T \Vdash f(u, v)$	$\rightsquigarrow$	$\mathcal{C} \wedge T \Vdash u \wedge T \Vdash v$	for $f \in \{\langle\rangle, \mathrm{enc}, \mathrm{enca}, \mathrm{sign}\}$

Theorem 2 *Let $\mathcal{C}$ be an unsolved constraint system.*

1. *(Correctness) If $\mathcal{C} \rightsquigarrow_\sigma^* \mathcal{C}'$ for some constraint system $\mathcal{C}'$ and some substitution σ and if θ is a solution of $\mathcal{C}'$ then $\sigma\theta$ is a solution of $\mathcal{C}$.*

2. *(Completeness) If θ is a solution of $\mathcal{C}$, then there exist a solved constraint system $\mathcal{C}'$ and substitutions σ, θ' such that $\theta = \sigma\theta'$, $C \rightsquigarrow_{\sigma}^{*} \mathcal{C}'$ and θ' is a solution of $\mathcal{C}'$.*

3. *(Termination) There is no infinite chain $\mathcal{C} \rightsquigarrow_{\sigma_1} \mathcal{C}_1 \ldots \rightsquigarrow_{\sigma_n} \mathcal{C}_n$.*

Proving that forbiding unification between pairs still leads to a complete decision procedure required in particular to introduce a new notion of minimality for tree proofs for deduction. Note that this result is of independent interest. Indeed, we provide a more efficient decision procedure for solving constraint systems, thus for deciding secrecy for a bounded number of sessions. Of course, the theoretical worst-case complexity remains the same (NP).

Proposition 1 is then proved in three main steps (see Appendix). First, Theorem 2 serves as a key result for proving that if $\mathcal{C}$ is satisfiable, then there exists a solution θ such that every term in $\mathcal{C}\theta$ is well-tagged. Intuitively, it shows that there is a solution where messages from Π_1 and Π_2 are not mixed up.

Second, conditions 1 and 2 ensure that for any solution θ of $\mathcal{C}$, the critical extended names of KC do not appear in plaintext in $\mathcal{C}\theta$.

Third, thanks to the two previous steps, we prove that β-tagged terms (intuitively messages from Π_2) are not useful for deducing α-tagged terms. The proof required in particular the introduction of a new locality lemma for deduction of ground terms. We deduce that, removing from $\mathcal{C}$ all constraints inherited from Π_2 and all β-tagged terms, we obtain a satisfiable constraint $\mathcal{C}'$ that is associated to a scenario of Π_1.

5 Conclusion

In this paper, we have shown how to safely compose secure protocols by tagging encryption, focusing on secrecy properties. Whenever a protocol preserves the secrecy of some data s, it still preserves s secrecy when other tagged protocols are executed in the same environment. We plan to consider the protocol composition problem for larger classes of security properties. In particular, we believe that our result can be extended to authentication-like properties.

More broadly, we foresee composition results in a more general way. In this paper, protocols are composed in the sense that they can be executed in the same environment. We plan to develop composition results where protocols can use other protocols as subprograms. For example, a protocol could use a secure channel, letting the implementation of the secure channel underspecified. This secure channel could be then possibly implemented by any protocol establishing session keys.

References

[1] M. Abadi and R. M. Needham. Prudent engineering practice for cryptographic protocols. *IEEE Trans. Software Eng.*, 22(1):6–15, 1996.

[2] R. Amadio and W. Charatonik. On name generation and set-based analysis in the Dolev-Yao model. In *Proc. Inter. Conference on Concurrency Theory (CONCUR'02)*, volume 2421 of *LNCS*, pages 499–514. Springer-Verlag, 2002.

[3] B. Blanchet. An efficient cryptographic protocol verifier based on Prolog rules. In *Proc. 14th Computer Security Foundations Workshop (CSFW'01)*, pages 82–96. IEEE Comp. Soc. Press, 2001.

[4] B. Blanchet and A. Podelski. Verification of cryptographic protocols: Tagging enforces termination. In *Foundations of Software Science and Computation Structures (FoSSaCS'03)*, volume 2620 of *LNCS*. Springer, 2003.

[5] R. Canetti, C. Meadows, and P. F. Syverson. Environmental requirements for authentication protocols. In *Proc. Symposium on Software Security – Theories and Systems*, volume 2609 of *LNCS*, pages 339–355. Springer, 2002.

[6] H. Comon-Lundh and V. Cortier. Security properties: two agents are sufficient. *Science of Computer Programming*, 50(1-3):51–71, 2004.

[7] H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *Proc. 18th Annual Symposium on Logic in Comp. Sc. (LICS'03)*, pages 271–280. IEEE Comp. Soc. Press, 2003.

[8] V. Cortier and E. Zalinescu. Deciding key cycles for security protocols. In *Proc. 13th Inter. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'06)*, volume 4246 of *LNCS*, pages 317–331. Springer, 2006.

[9] A. Datta, A. Derek, J. Mitchell, and D. Pavlovic. A derivation system and compositional logic for security protocols. *Journal of Computer Security*, 13(3), 2005.

[10] A. Datta, A. Derek, J. C. Mitchell, and A. Roy. Protocol composition logic (PCL). *Electr. Notes Theor. Comput. Sci.*, 172:311–358, 2007.

[11] L. Gong and P. Syverson. Fail-stop protocols: An approach to designing secure protocols. In *Proc. 5th Inter. Working Conference on Dependable Computing for Critical Applications*, pages 44–55, 1995.

[12] J. D. Guttman and F. J. Thayer. Protocol independence through disjoint encryption. In *Proc. 13th Computer Security Foundations Workshop (CSFW'00)*, pages 24–34. IEEE Comp. Soc. Press, 2000.

[13] J. Kelsey, B. Schneier, and D. Wagner. Protocol interactions and the chosen protocol attack. In *Proc. 5th Inter. Workshop on Security Protocols*, volume 1361 of *LNCS*, pages 91–104. Springer, 1997.

[14] G. Lowe. Casper: A compiler for the analysis of security protocols. In *Proc. 10th Computer Security Foundations Workshop (CSFW'97)*. IEEE Comp. Soc. Press, 1997.

[15] J. K. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *Proc. 8th ACM Conference on Computer and Communications Security (CCS'01)*, pages 166–175, 2001.

[16] R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communication of the ACM*, 21(12):993–999, 1978.

[17] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions and composed keys is NP-complete. *Theoretical Comp. Sc.*, 299:451–475, 2003.

[18] H. Seidl and K. N. Verma. Flat and one-variable clauses: Complexity of verifying cryptographic protocols with single blind copying. In *Proc. 11th Inter. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'04)*, volume 3452 of *LNCS*. Springer, 2005.

A Proof of completeness

Let $T_1 \subseteq T_2 \subseteq \ldots \subseteq T_n$. We say that a proof π of $T_i \vdash u$ is *left-minimal* if for any $j < i$ such that $T_j \vdash u$, π' is a proof of $T_j \vdash u$ where π' is obtained from π by replacing T_i with T_j in the left hand side of each node of π.

Definition 8 (simple) *We say that a proof π is* simple *if*

- *any subproof of π is left-minimal,*
- *any term of the form $\langle u_1, u_2\rangle$ obtained by application of a decomposition rule or an axiom rule is followed by a projection rule*
- *a composition rule is not directly followed by a decomposition rule.*

Example 4 *Let $T_1 = \{a\}$ and $T_2 = \{a, \mathrm{enc}(\langle a, b\rangle, k), k\}$. We have that $T_2 \vdash \langle a, b\rangle$.*

$$\dfrac{\dfrac{}{T_2 \vdash \mathrm{enc}(\langle a, b\rangle, k)} \quad \dfrac{}{T_2 \vdash k}}{T_2 \vdash \langle a, b\rangle}$$

The proof above is not a simple proof of $T_2 \vdash \langle a, b\rangle$. The term $\langle a, b\rangle$ has been obtained by an application of a decomposition rule. Thus we have to decompose it. A simple proof of $T_2 \vdash \langle a, b\rangle$ is described below:

$$\dfrac{\dfrac{\dfrac{}{T_2 \vdash \mathrm{enc}(\langle a, b\rangle, k)} \quad \dfrac{}{T_2 \vdash k}}{T_2 \vdash \langle a, b\rangle} \quad \dfrac{}{T_2 \vdash a}}{T_2 \vdash b}$$

$$\dfrac{\dfrac{\dfrac{}{T_2 \vdash \mathrm{enc}(\langle a, b\rangle, k)} \quad \dfrac{}{T_2 \vdash k}}{T_2 \vdash \langle a, b\rangle} \quad \dfrac{\dfrac{}{T_2 \vdash a}}{T_2 \vdash b}}{T_2 \vdash \langle a, b\rangle}$$

Lemma 1 *If $T_i \vdash u$ then there is a simple proof of it.*

Given a constraint system $\mathcal{C}$, we say that T_i is a minimal unsolved left hand side of $\mathcal{C}$ if T_i is a left hand side of $\mathcal{C}$ and for all $T \Vdash u \in \mathcal{C}$ such that $T \subsetneq T_i$, u is a variable.

Lemma 2 *Let $\mathcal{C}$ be an unsolved constraint system, θ be a solution to $\mathcal{C}$ and T_i be a minimal unsolved left hand side of $\mathcal{C}$. If there is a simple proof of $T_i\theta \vdash u$ having the last rule an axiom or a decomposition then there is $t \in St(T_i) \setminus \mathcal{X}$ such that $t\theta = u$.*

Proof. Consider a simple proof π of $T_i\theta \vdash u$. Let j be minimal such that the proof π' obtained from π by replacing T_i with T_j is a proof of $T_j\theta \vdash u$. According to the last applied rule in the proof, we have:

- The last rule is an axiom.
  Then $u \in T_j\theta$ and hence there is $t \in T_j$ such that $t\theta = u$. If t is a variable then $T_t \Vdash t$ is a constraint in $\mathcal{C}$ with $T_t \subsetneq T_j$ (see the definition of a constraint system). Hence $T_t\theta \vdash t\theta$, that is $T_t\theta \vdash u$, which contradicts the minimality of j.

- The last rule is a decomposition.
  Suppose that it is a symmetric decryption. Then, in such a case, there exists w such that $T_j\theta \vdash \mathrm{enc}(u, w)$ and $T_j\theta \vdash w$. By simplicity of the proof, the last rule applied to obtain $\mathrm{enc}(u, w)$ can not be a composition. Hence, it is either an axiom or a decomposition. Then, applying the induction hypothesis we have that there is $t \in St(T_j)$, t not a variable, such that $t\theta = \mathrm{enc}(u, w)$. It follows that $t = \mathrm{enc}(t', t'')$ with $t'\theta = u$. If t' is a variable then $T_{t'}\theta \vdash t'\theta$, that is $T_{t'}\theta \vdash u$ which contradicts the minimality of j. Hence t' is not a variable. For the other decomposition rules, the same reasoning holds. □

Let t be a term, we denote by $\mathsf{comp}(t)$ the components of the term t. This notion is formally defined as follows: $\mathsf{comp}(\langle t_1, t_2\rangle) = \mathsf{comp}(t_1) \cup \mathsf{comp}(t_2)$ and $\mathsf{comp}(t) = t$ otherwise.

Lemma 3 *Let $\mathcal{C}$ be an unsolved constraint system, θ be a solution of $\mathcal{C}$ and T_i be a minimal unsolved left hand side of $\mathcal{C}$ such that for all $t_1, t_2 \in St(T_i)$*

$$t_1\theta = t_2\theta \quad \textit{implies} \quad t_1 \textit{ or } t_2 \textit{ is a variable or a pair}$$

If $u_i \in St(T_i) \setminus \mathcal{X}$ and $T_i\theta \vdash u_i\theta$ then $T_i' \vdash u_i$ where $T_i' = T_i \cup \{x \mid T \Vdash x \in \mathcal{C}, T \subsetneq T_i\}$.

Proof. Let j be minimal such that $T_j\theta \vdash u_i\theta$. Thus $j \leq i$ and $T_j \subseteq T_i$. Consider a simple proof of $T_j\theta \vdash u_i\theta$. We reason by induction on the depth of the proof. We can have that:

- The proof is reduced to an application of the rule axiom that might be followed by several application of the projection rules until the resulting term is not a pair. Since the proof is a simple proof, we have that $u_i\theta$ is not a pair. Hence, u_i is not a pair.

  There exists $t \in T_j$ such that $u_i\theta \in \mathsf{comp}(t\theta)$. Either $u_i\theta = t'\theta$ for some $t' \in \mathsf{comp}(t) \setminus \mathcal{X}$ or $u_i\theta \in \mathsf{comp}(x\theta)$ for some $x \in \mathsf{comp}(t) \cap \mathcal{X}$. In the first case, we easily deduce that neither u_i nor t is a pair or a variable and hence by hypothesis, we have that $u_i = t'$ and hence $T_i' \vdash u_i$. In the second case, we have that $T_x\theta \vdash x\theta$. Thus $T_x\theta \vdash u_i\theta$ which contradicts the minimality of j, since $T_x \subsetneq T_j$.

- The proof ends with an application of a decomposition rule that might be followed by several application of the projection rules until the resulting term is not a pair. Note that, since the proof is a simple proof, we have that $u_i\theta$ is not a pair. Hence u_i is not a pair.

  Suppose for example that it is the symmetric decryption rule. That is, there exist w_1, w_2 such that $T_j\theta \vdash \mathrm{enc}(w_1, w_2)$, $T_j\theta \vdash w_2$ and $u_i\theta \in \mathsf{comp}(w_1)$. The last rule applied to obtain $T_j\theta \vdash \mathrm{enc}(w_1, w_2)$ was not a composition by simplicity of the proof.

We can hence apply Lemma 2 and obtain that there is $t \in St(T_j) \setminus \mathcal{X}$ such that $t\theta = \text{enc}(w_1, w_2)$. Since t is not a variable, we have that $t = \text{enc}(t_1, t_2)$ with $t_1\theta = w_1$ and $t_2\theta = w_2$. Either $u_i\theta = p\theta$ for some $p \in \mathsf{comp}(t_1) \setminus \mathcal{X}$ or $u_i\theta \in \mathsf{comp}(x\theta)$ for some $x \in \mathsf{comp}(t_1) \cap \mathcal{X}$. In the second case, we have that $T_x\theta \vdash x\theta$. Thus $T_x\theta \vdash u_i\theta$ which contradicts the minimality of j, since $T_x \subsetneq T_j$. In the first case, we easily deduce that neither u_i nor p is a variable or a pair and hence by hypothesis, we have that $u_i = p$. We can apply the induction hypothesis on $T_j\theta \vdash \text{enc}(t_1, t_2)\theta$ (this subproof is simple) to obtain that $T'_i \vdash \text{enc}(t_1, t_2)$.

Now, it t_2 is a variable then $t_2 \in T'_i$, thus $T'_i \vdash t_2$. Otherwise, if t_2 is not a variable then, by induction hypothesis on $T_j\theta \vdash t_2\theta$ (this subproof is a simple one), we obtain $T'_i \vdash t_2$. Hence, in both cases, we obtain that $T'_i \vdash t_2$. Then, together with $T'_i \vdash \text{enc}(t_1, t_2)$ and $u_i \in \mathsf{comp}(t_1)$, it follows that $T'_i \vdash u_i$. For the other decomposition rules the same reasoning holds.

- The last rule is a composition.
  Suppose that it is the symmetric encryption rule. Then $u_i\theta = \text{enc}(w_1, w_2)$ and $T_j\theta \vdash w_1$ and $T_j\theta \vdash w_2$. Since u_i is not a variable, we have that $u_i = \text{enc}(v'_1, v'_2)$, $v'_1\theta = w_1$ and $v'_2\theta = w_2$. If v'_1 (resp. v'_2) is a variable then v'_1 (resp. v'_2) is in T'_i (this is because $v_j \in St(T_i)$). Otherwise, we apply our induction hypothesis (note that the two subproofs are simple). Hence, in both cases, we have that $T'_i \vdash v'_1$ and also that $T'_i \vdash v'_2$. Hence, we easily deduce that $T'_i \vdash u_i$. For the other composition rules the same reasoning holds. □

Proposition 2 (completeness) *Let $\mathcal{C}$ be an unsolved constraint system and θ be a solution of $\mathcal{C}$. Then, there is a constraint system $\mathcal{C}'$ and a solution τ of $\mathcal{C}'$ such that $\mathcal{C} \rightsquigarrow_\sigma \mathcal{C}'$ and $\theta = \sigma\tau$.*

Proof. Consider the minimal unsolved constraint $T_i \Vdash u_i$. Hence, we have that u_i is not a variable whereas u_j is a variable for all $j < i$. Firstly, assume that $u_i = \langle v_1, v_2 \rangle$ for some terms v_1, v_2. In such a case, let $\mathcal{C}'$ be the constraint system obtained from $\mathcal{C}$ by applying $\mathsf{R}_{\langle\rangle}$ and $\tau = \theta$. Since $T_i\theta \vdash u_i\theta$, we have also that $T_i\theta \vdash v_1\theta$ and $T_i\theta \vdash v_2\theta$ meaning that $\tau = \theta$ is a solution of $\mathcal{C}'$.

Now, assume that u_i is neither a variable nor a pair and consider a simple proof of $T_i\theta \vdash u_i\theta$. According to the last applied rule in this proof, we have:

1. The last rule is a composition.
   Suppose that it is the symmetric encryption rule. Hence, there are w_1, w_2 such that $T_i\theta \vdash w_1$ and $T_i\theta \vdash w_2$ and $\text{enc}(w_1, w_2) = u_i\theta$. Since u_i is not a variable, there exist v_1, v_2 such that $u_i = \text{enc}(v_1, v_2)$. Let $\mathcal{C}'$ be the constraint system obtained from $\mathcal{C}$ by applying the simplification rule $\mathsf{R}_{\mathsf{enc}}$ on $T \Vdash \text{enc}(v_1, v_2)$. Since $v_1\theta = w_1$ and $v_2\theta = w_2$, the substitution θ is also a solution to $\mathcal{C}'$. For the other composition rules the same reasoning holds, applying this time the corresponding R_f rule.

2. The last rule is an axiom or a decomposition.
   Applying Lemma 2 we obtain that there is $t \in St(T_i) \setminus \mathcal{X}$ such that $t\theta = u_i\theta$. We distinguish two cases:

   - $t \neq u_i$. Note that u_i is neither a pair nor a variable. Since $t\theta = u_i\theta$ and t is not a variable, we easily deduce that t is not a pair. Hence, we can apply the simplification rule R_2.
   - $t = u_i$. In such a case, we have that $u_i \in St(T_i)$. Either there are two distinct non variable and non pair terms $t_1, t_2 \in St(T_i)$ such that $t_1\theta = t_2\theta$ and we apply the simplification rule R_3. Otherwise, the simplification rule R_1 can be applied. This follows from Lemma 3. □

B Proof of our combination result

The *left-hand side* of a constraint system $\mathcal{C}$, denoted by $\mathsf{lhs}(\mathcal{C})$, is the maximal left-hand side of the constraints of $\mathcal{C}$. The *right-hand side* of a constraint system $\mathcal{C}$, denoted by $\mathsf{rhs}(\mathcal{C})$, is the set of right-hand sides of its constraints. The set $vars(\mathcal{C})$ denotes the set of variables occurring in $\mathcal{C}$ and $\bot$ denotes the unsatisfiable system.

B.1 Existence of a solution without any mixing

Consider a constraint system $\mathcal{C}$ issued from $\Pi_1 \mid \Pi_2$. The goal of this subsection is to establish the existence of a solution of $\mathcal{C}$ having some features (see Proposition 3).

Lemma 4 *Let T_α and T_β be two sets of terms which are respectively α-tagged and β-tagged and such that $vars(T_\alpha) \cap vars(T_\beta) = \emptyset$. Let $\mathcal{C}$ be a constraint system such that $\mathsf{lhs}(\mathcal{C}) \cup \mathsf{rhs}(\mathcal{C}) \subseteq St(T_\alpha \cup T_\beta)$. Let $\mathcal{C}'$ be a constraint system such that $\mathcal{C} \rightsquigarrow_\sigma^* \mathcal{C}'$ for some substitution σ. We have that $\mathsf{lhs}(\mathcal{C}') \cup \mathsf{rhs}(\mathcal{C}') \subseteq St(T_\alpha\sigma \cup T_\beta\sigma)$. Moreover, terms in $T_\alpha\sigma$ and $T_\beta\sigma$ are respectively α-tagged and β-tagged, $\mathrm{n}(T_\alpha\sigma) \subseteq \mathrm{n}(T_\alpha)$, $\mathrm{n}(T_\beta\sigma) \subseteq \mathrm{n}(T_\beta)$ and $vars(T_\alpha\sigma) \cap vars(T_\beta\sigma) = \emptyset$.*

Proof. This result is easy to prove by induction on the length ℓ of the derivation from $\mathcal{C}$ to $\mathcal{C}'$. When $\ell = 0$, the result is obvious. Now, assume that $\ell \geq 1$. In such a case, we have that there exists $\mathcal{C}_1$, σ_1 and σ_2 such that $\mathcal{C} \rightsquigarrow_{\sigma_1} \mathcal{C}_1 \rightsquigarrow_{\sigma_2}^* \mathcal{C}'$ and $\sigma = \sigma_1\sigma_2$. If the rule involved in the first step is R_1, R_4 or R_5, then we easily conclude by applying the induction hypothesis. Now, assume that the simplification rule involved in the first step is either R_2 or R_3. In such a case, we know that $\sigma_1 = \mathrm{mgu}(t_1, t_2)$ with $t_1, t_2 \in St(T_\alpha \cup T_\beta)$ Actually, it is impossible that $t_1 \in St(T_\alpha)$ and $t_2 \in St(T_\beta)$ (or the converse). Indeed, t_1, t_2 are neither variables, nor pairs and have to contain the same tag to be unifiable. Hence, we know that $t_1, t_2 \in St(T_\alpha)$ (or $t_1, t_2 \in St(T_\beta)$). Now, it is easy to see that σ_1 is α-tagged (or β-tagged), thus $T_\alpha\sigma_1$ (or $T_\beta\sigma_1$) too. Since $vars(T_\alpha) \cap vars(T_\beta) = \emptyset$, we have that $T_\beta\sigma_1 = T_\beta$ (or $T_\alpha\sigma_1 = T_\alpha$). Moreover, it is easy to see that $\mathrm{n}(T_\alpha\sigma_1) \subseteq \mathrm{n}(T_\alpha)$ and $\mathrm{n}(T_\beta\sigma_1) \subseteq \mathrm{n}(T_\beta)$. Then, we can apply our induction hypothesis on $\mathcal{C}_1 \rightsquigarrow_{\sigma_2}^* \mathcal{C}'$. Putting all together we easily conclude. □

Lemma 5 *Let T be a set of ground terms and u be a ground term such that $T \vdash u$. Then, we have that $\mathit{plaintext}(u) \subseteq \mathit{plaintext}(T)$.*

Proof. let π be a proof of $T \vdash u$. We prove this result by induction on the depth of π. We can have:

- The last rule is an axiom. Then $u \in T$ and we have that $\mathit{plaintext}(u) \subseteq \mathit{plaintext}(T)$.
- The last rule is a composition. Suppose for example that it is the symmetric encryption rule. Then $u = \mathrm{enc}(u_1, u_2)$, $T \vdash u_1$ and $T \vdash u_2$. By definition, we have that $\mathit{plaintext}(u) = \mathit{plaintext}(u_1)$. Hence, we easily conclude by applying our induction hypothesis on $T \vdash u_1$.
- The last rule is a decomposition. Suppose for example that it is the symmetric decryption rule. In such a case, we have that $T \vdash \mathrm{enc}(u, v)$ and $T \vdash v$ for some term v. By induction hypothesis, $\mathit{plaintext}(\mathrm{enc}(u, v)) \subseteq \mathit{plaintext}(T)$. Hence, we easily conclude that $\mathit{plaintext}(u) \subseteq \mathit{plaintext}(T)$. □

Definition 9 *Let $\mathcal{C}$ be a constraint system. We say that $\mathcal{C}$ satisfies the* origination property *if the following holds*

if $x \in \mathit{plaintext}(T) \cap \mathcal{X}$ for some $(T \Vdash u) \in \mathcal{C}$ then

$$T_x^p \stackrel{\mathsf{def}}{=} \min\{T' \mid (T' \Vdash u') \in \mathcal{C} \text{ and } x \in \mathit{plaintext}(u')\}$$

exists and $T_x^p \subsetneq T$.

Intuitively, this means that when a variable x appears at a plaintext position on the left-hand side of a constraint, then x appears also at a plaintext position on the right-hand side of a smaller constraint (*w.r.t.* the inclusion ordering of the left-hand sides of the constraints)

Proposition 3 *Let T_0 and KC be two sets of extended names such that $\mathsf{init} \in T_0$. Let T_α and T_β be two sets of terms which are respectively α-tagged and β-tagged and such that $\mathit{vars}(T_\alpha) \cap \mathit{vars}(T_\beta) = \emptyset$ and $(\mathit{plaintext}(T_\alpha) \cup \mathit{plaintext}(T_\beta)) \cap \mathsf{KC} = \emptyset$.*

Let $\mathcal{C}$ be a constraint system such that $\mathsf{lhs}(\mathcal{C}) \subseteq T_0 \cup T_\alpha \cup T_\beta$, $\mathsf{rhs}(\mathcal{C}) \subseteq T_\alpha \cup T_\beta$ and which satisfies the origination property. If $\mathcal{C}$ is satisfiable, then there exists a solution θ' of $\mathcal{C}$ such that

1. *$T_\alpha\theta'$ is a set of α-tagged terms and $\mathrm{n}(T_\alpha\theta') \subseteq \mathrm{n}(T_\alpha) \cup \{\mathsf{init}\}$,*
2. *$T_\beta\theta'$ is a set of β-tagged terms and $\mathrm{n}(T_\beta\theta') \subseteq \mathrm{n}(T_\beta) \cup \{\mathsf{init}\}$,*
3. *for all $x \in \mathit{plaintext}(T_\alpha \cup T_\beta)$, we have that $\mathit{plaintext}(x\theta') \cap \mathsf{KC} = \emptyset$.*

Proof. Let T_0, T_α, T_β and $\mathcal{C}$ be as described in the proposition and let θ be a solution of $\mathcal{C}$. Thanks to our completeness result (Theorem 2), we know that there exists a constraint system $\mathcal{C}'$ in solved form and a substitution σ such that $\mathcal{C} \rightsquigarrow_\sigma^* \mathcal{C}'$. Let $\theta' = \sigma\tau$ where $x\tau = \mathsf{init}$ for every $x \in vars(\mathcal{C}')$. It is clear that θ' is a solution of $\mathcal{C}$, it remains to show that θ' satisfies the requirements.

Thanks to Lemma 4, it is easy to establish that

- terms in $T_\alpha\sigma$ are α-tagged and also that $\mathrm{n}(T_\alpha\sigma) \subseteq \mathrm{n}(T_\alpha)$,
- terms in $T_\beta\sigma$ are β-tagged and also that $\mathrm{n}(T_\beta\sigma) \subseteq \mathrm{n}(T_\beta)$.

From this, it is easy to establish the two first conditions.

- terms in $T_\alpha\theta' = T_\alpha\sigma\tau$ are α-tagged and $\mathrm{n}(T_\alpha\theta') \subseteq \mathrm{n}(T_\alpha) \cup \{\mathsf{init}\}$,
- terms in $T_\beta\theta' = T_\beta\sigma\tau$ are β-tagged and $\mathrm{n}(T_\beta\theta') \subseteq \mathrm{n}(T_\beta) \cup \{\mathsf{init}\}$.

Now, let $V = \{x \in plaintext(T_\alpha \cup T_\beta) \cap \mathcal{X} \mid plaintext(x\theta') \cap \mathsf{KC} \neq \emptyset\}$. If $V = \emptyset$, then the condition holds. Otherwise, let $x \in V$ such that T_x^p is minimal *w.r.t.* the inclusion ordering (see Definition 9 for the definition of T_x^p). Since $\mathcal{C}$ satisfies the origination property, there exists $(T_x^p \Vdash u) \in \mathcal{C}$ such that $x \in plaintext(u)$ and $x \notin plaintext(T_x^p)$. We have that $plaintext(u\theta') \cap \mathsf{KC} \neq \emptyset$ and thanks to Lemma 5, we deduce that $plaintext(T_x^p\theta) \cap \mathsf{KC} \neq \emptyset$. It is easy to see that $plaintext(T_x^p) \cap \mathsf{KC} = \emptyset$ since this property holds for T_α, T_β and T_0. Hence, we deduce that there exists $y \in vars(T_x^p)$ such that $plaintext(y\theta) \cap \mathsf{KC} \neq \emptyset$ and $y \in plaintext(T_x^p)$. This means that there exists a variable $y \in V$ such that $T_y^p \subsetneq T_x^p$, contradiction. □

B.2 Getting rid of the terms coming from Π_2

We define a function, denoted by $\overline{\cdot}$, whose goal is to project terms which come from Π_2 (and which are β-tagged) onto a special term init. Given a set Names of names, we define the function $\overline{\cdot}$ inductively as follows:

- $\overline{u} = \mathsf{init}$ if $u \in \mathsf{Names}$,
- $\overline{u} = u$ if u is a name and $u \notin \mathsf{Names}$,
- $\overline{f(\langle\beta, u_1\rangle, u_2)} = \mathsf{init}$ if $f \in \{\mathrm{enca}, \mathrm{enc}, \mathrm{sign}\}$
- $\overline{f(u_1, \ldots, u_n)} = f(\overline{u_1}, \ldots, \overline{u_n})$ otherwise

Before to prove Lemmas 7 and 8 which will be useful to establish our main result (Proposition 1), we need to show a locality lemma (Lemma 6). This locality lemma relies on the following definition.

Definition 10 ($St_{plain}(t)$) *Let t be a ground term. The set $St_{plain}(t)$ of subterms of t that appear at a plaintext position is inductively defined as follows:*

- $St_{plain}(u) = \{u\}$ *if* $u \in \mathrm{n}(u)$
- $St_{plain}(f(u_1, u_2)) = \{f(u_1, u_2)\} \cup St_{plain}(u_1)$ *if* $f \in \{\mathrm{enc}, \mathrm{enca}, \mathrm{sign}\}$
- $St_{plain}(\langle u_1, u_2\rangle) = \{\langle u_1, u_2\rangle\} \cup St_{plain}(u_1) \cup St_{plain}(u_2)$.

Lemma 6 (locality) *Let T be a set of ground terms and u be a ground term such that $T \vdash u$. Let π be a proof of $T \vdash u$ which is minimal w.r.t. its number of nodes. Then π only involves terms in $St(T, u)$. Moreover, if π ends with an instance of a decomposition rule or an instance of the axiom rule then π only involves terms in $St(T)$ and $u \in St_{plain}(T)$.*

Proof. Let π be a proof of $T \vdash u$ which is minimal *w.r.t.* to its number of nodes. We will show the result by induction on π. We can have that:

- The last rule is an axiom. In such a case, we easily conclude.
- The last rule is a composition. Suppose for example that it is the symmetric encryption rule. In such a case, we have that $u = \mathrm{enc}(u_1, u_2)$. Let π_1 (resp. π_2) be the subproof of π ending on $T \vdash u_1$ (resp. $T \vdash u_2$). By induction hypothesis, we know that π_1 (resp. π_2) only involves terms in $St(T, u_1)$ (resp. $St(T, u_2)$). Hence, we easily deduce that π only involves terms in $St(T, u)$. The same reasoning holds for the other composition rules.
- The last rule is a decomposition. Suppose for example that it is the symmetric decryption rule. In such a case, we have that

$$\dfrac{\pi_1 = \left\{ \dfrac{\cdots}{T \vdash \mathrm{enc}(u, v)} \right. \qquad \pi_2 = \left\{ \dfrac{\cdots}{T \vdash v} \right.}{T \vdash u}$$

  Note that, by minimality of π, the proof π_1 necessarily ends with a decomposition rule. Hence, by induction hypothesis, we know that π_1 only involves terms in $St(T)$ and also that $\mathrm{enc}(u, v) \in St_{plain}(T)$. Then, we easily deduce that π only involves terms in $St(T)$ and also that $u \in St_{plain}(T)$. For the other decomposition rules a similar reasoning holds. In the case of the asymmetric decryption rule, we have that $v \in St(T)$ since a term of the form $\mathrm{priv}(v')$ can only be obtained by the axiom rule or an instance of a decomposition rule. □

Lemma 7 *Let T_0 be a set of extended names and* Names *be a set of names such that $\mathrm{n}(T_0) \cap$* Names $= \emptyset$ *and* init $\in T_0$. *Let v be a β-tagged term such that plaintext$(v) \subseteq T_0 \cup$* Names. *Then, we have that $T_0 \vdash \overline{v}$.*

The proof below relies on the notion of component which is formally defined in Appendix A.
Proof. We will show that for every $p \in \mathsf{comp}(v)$, we have that $T_0 \vdash \overline{p}$. By definition of $\overline{\cdot}$, we have that

$$\{\overline{p} \mid p \in \mathsf{comp}(v)\} = \{p' \mid p' \in \mathsf{comp}(\overline{v})\}.$$

Then, we easily deduce that $T_0 \vdash p'$ for every $p' \in \mathsf{comp}(\overline{v})$, and thus $T_0 \vdash \overline{v}$.

Let $p \in \mathsf{comp}(v)$. We distinguish three cases:

1. p is of the form $\mathrm{enc}(w_1, w_2)$, $\mathrm{enca}(w_1, w_2)$ or $\mathrm{sign}(w_1, w_2)$. In such a case, since p is β-tagged, we have that $\overline{p} = \mathsf{init}$, thus $T_0 \vdash \overline{p}$.

2. p is of the form $\mathrm{pub}(t)$ (or $\mathrm{priv}(t)$), thus $\mathrm{pub}(t) \in T_0 \cup \mathsf{Names}$. We have that $p \in T_0$ and thus $\overline{p} \in T_0$ since $\mathrm{n}(T_0) \cap \mathsf{Names} = \emptyset$.

3. p is a name. We have that $p \in \mathit{plaintext}(v)$ and $p \in T_0 \cup \mathsf{Names}$, thus $T_0 \vdash \overline{p}$. This allows us to conclude. □

Remark. The condition $T_0 \cap \mathsf{Names} = \emptyset$ is not sufficient to prove Lemma 7. For instance, let $v = \mathrm{pub}(a)$, $\mathsf{Names} = \{a\}$ and $T_0 = \{\mathrm{pub}(a)\}$. We have that $\overline{v} = \mathrm{pub}(\mathsf{init})$ and $\overline{v}$ is not deducible from T_0.

Lemma 8 *Let T_0 be a set of extended Names and Names be a set of names such that $\mathrm{n}(T_0) \cap \mathsf{Names} = \emptyset$ and $\mathsf{init} \in T_0$. Let T_α be a set of α-tagged terms and T_β be a set of β-tagged terms such that $\mathit{plaintext}(T_\beta) \subseteq T_0 \cup \mathsf{Names}$. Let u be a ground term such that $T_0, T_\alpha, T_\beta \vdash u$. Then we have that $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \overline{u}$.*

Proof. By hypothesis, we have $T_0, T_\alpha, T_\beta \vdash u$. Let π be a proof of $T_0, T_\alpha, T_\beta \vdash u$ which is minimal *w.r.t.* its number of nodes. We will show that $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \overline{u}$ by induction on the proof. We can have that:

- The last rule is an axiom. In such a case, we have that $u \in T_0 \cup T_\alpha \cup T_\beta$. We easily deduce that $\overline{u} \in \overline{T_0} \cup \overline{T_\alpha} \cup \overline{T_\beta}$. This allows us to conclude that $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \overline{u}$.

- The last rule is a composition. Either $\overline{u} = \mathsf{init}$ and we easily conclude. Otherwise, suppose for example that the last rule is the symmetric encryption rule. In such a case, we have that $u = \mathrm{enc}(u_1, u_2)$ and $\overline{u} = \mathrm{enc}(\overline{u_1}, \overline{u_2})$. By induction hypothesis, we know that $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \overline{u_1}$ and $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \overline{u_2}$. Hence, we deduce that $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \mathrm{enc}(\overline{u_1}, \overline{u_2})$, that is $\overline{T_0}, \overline{T_\alpha}, \overline{T_\beta} \vdash \overline{u}$

- The last rule is a decomposition. Suppose for example that it is the symmetric decryption rule. In such a case, we have that

$$\dfrac{\pi_1 = \left\{ \dfrac{\cdots}{T_0, T_\alpha, T_\beta \vdash \mathrm{enc}(u, v)} \right. \quad \pi_2 = \left\{ \dfrac{\cdots}{T_0, T_\alpha, T_\beta \vdash v} \right.}{T_0, T_\alpha, T_\beta \vdash u}$$

  If u is not of the form $\langle \beta, u_1 \rangle$ for some term u_1, then by applying our induction hypothesis, we easily conclude since $\overline{\mathrm{enc}(u, v)} = \mathrm{enc}(\overline{u}, \overline{v})$.

Now, we have to consider the case where u is of the form $\langle \beta, u_1 \rangle$. By minimality of the proof we know that π_1 ends either with an instance of axiom or with an instance of a decomposition rule. Hence, by Lemma 6, we have that $\mathrm{enc}(u, v) \in St_{plain}(T_0 \cup T_\alpha \cup T_\beta)$. Moreover, since $\mathrm{enc}(u, v)$ is β-tagged, we finally deduce that $\mathrm{enc}(u, v) \in St_{plain}(T_\beta)$ and hence $u \in St_{plain}(T_\beta)$. Since $plaintext(T_\beta) \subseteq T_0 \cup \mathsf{Names}$, we easily deduce that $plaintext(u) \subseteq T_0 \cup \mathsf{Names}$ and Lemma 7 allows us to conclude that $T_0 \vdash \overline{u}$.

For the other decomposition rules, a similar reasoning holds. □

B.3 Proof of the Proposition 1

Proposition 1 *Let $\Pi_1 = [k_1] \to \mathsf{Roles}$, $\Pi_2 = [k_2] \to \mathsf{Roles}$, T_0 and m defined as in Theorem 1 and satisfying the conditions 1 and 2. Let $k = k_1 + k_2$ and sc be a scenario for $\Pi_1 \mid \Pi_2$. For any role number $1 \leq i \leq k$, let $s_i \in \mathbb{N}$ such that $(i, s_i) \in \mathsf{sc}$ or $\forall j, (j, s_i) \notin \mathsf{sc}$. Let $\mathcal{C}$ be the constraint system associated to sc, T_0 and $m\sigma_{1,s_1} \cdots \sigma_{k,s_k}$. Let $\mathsf{sc}' = \mathsf{sc}|_{\Pi_1}$ and $\mathcal{C}'$ be the constraint system associated to sc', T_0 and $m\sigma_{1,s_1} \cdots \sigma_{k_1,s_{k_1}}$. If $\mathcal{C}$ is satisfiable, then $\mathcal{C}'$ is also satisfiable.*

Proof. Let $\Pi_1 : [k_1] \to \mathsf{Roles}$, $\Pi_2 : [k_2] \to \mathsf{Roles}$, T_0 and m defined as in Theorem 1. Let $k = k_1 + k_2$ and sc be a scenario for $\Pi_1 \mid \Pi_2$. For any role number $1 \leq i \leq k$, let $s_i \in \mathbb{N}$ such that $(i, s_i) \in \mathsf{sc}$ or $\forall j, (j, s_i) \notin \mathsf{sc}$. Let $(u_1, v_1) \cdots (u_n, v_n)$ be the sequence of rules associated to sc. Let $\mathcal{C}$ be the constraint system associated to sc, T_0 and $m_1 = m\sigma_{1,s_1} \cdots \sigma_{k,s_k}$. Let $\mathsf{sc}' = \mathsf{sc}|_{\Pi_1}$, *i.e.* the sequence obtained from sc by removing any element (r, s) where r is a role of Π_2. Let $\mathcal{C}'$ be the constraint system associated to sc', T_0 and the term $m_2 = m\sigma_{1,s_1} \cdots \sigma_{k_1,s_{k_1}}$. Note that $m_1 = m_2$ since $vars(m) \subseteq vars(\Pi_1)$. In the remainder, we will denote it by m'. For sake of simplicity, we will assume that $\mathsf{init} \in T_0$. The constraint systems $\mathcal{C}$ and $\mathcal{C}'$ are as follows:

$$\mathcal{C} := \left\{ \begin{array}{lcl} T_0 & \Vdash & u_1 \\ T_0, v_1 & \Vdash & u_2 \\ T_0, v_1, v_2 & \Vdash & u_3 \\ \cdots & \Vdash & \cdots \\ T_0, v_1, \ldots, v_n & \Vdash & m' \end{array} \right. \qquad \mathcal{C}' := \left\{ \begin{array}{lcl} T_0 & \Vdash & u_{i_1} \\ T_0, v_{i_1} & \Vdash & u_{i_2} \\ T_0, v_{i_1}, v_{i_2} & \Vdash & u_{i_3} \\ \cdots & \Vdash & \cdots \\ T_0, v_{i_1}, \ldots, v_{i_n} & \Vdash & m' \end{array} \right.$$

where $i_1 \cdots i_n$ is a sequence obtained from $1 \cdots n$ by removing any element j when the j^{th} element, say (r, s), of the sequence sc is such that $k_1 < r \leq k$. Intuitively, we remove the elements corresponding to a step of the protocol Π_2.

Now, before to apply Proposition 3, we have to check that all the hypothesis are satisfied. Let

- $T_\alpha = \{u_{i_1}, v_{i_1}, \ldots, u_{i_n}, v_{i_n}, m'\}$, and
- $T_\beta = \{u_j, v_j \mid 1 \leq j \leq n \text{ and } j \notin \{i_1, \ldots, i_n\}\}$.

First of all, we have that T_α and T_β are two sets of terms which are respectively α-tagged and β-tagged. We have that $vars(T_\alpha) \cap vars(T_\beta) = \emptyset$. Intuitively, this is because, terms in T_α come from Π_1 whereas terms in T_β come from Π_2. We have also that $\mathsf{lhs}(\mathcal{C}) \subseteq T_0 \cup T_\alpha \cup T_\beta$ and $\mathsf{rhs}(\mathcal{C}) \subseteq T_\alpha \cup T_\beta$. Moreover, $\mathcal{C}$ satisfies the origination property thanks to the condition 2 given in Theorem 1 and by hypothesis, we know that $\mathcal{C}$ is satisfiable. Hence, we can apply Proposition 3 to deduce that there exists a solution θ of $\mathcal{C}$ such that:

- $T_\alpha\theta$ is a set of α-tagged terms and $\mathrm{n}(T_\alpha\theta) \subseteq \mathrm{n}(T_\alpha) \cup \{\mathsf{init}\}$, $(\star)$
- $T_\beta\theta$ is a set of β-tagged terms and $\mathrm{n}(T_\beta\theta) \subseteq \mathrm{n}(T_\beta) \cup \{\mathsf{init}\}$, $(\star\star)$
- for all $x \in plaintext(T_\alpha \cup T_\beta)$, we have that $plaintext(x\theta) \cap \mathsf{KC} = \emptyset$.

Let $\theta' = \theta|_{vars}(\mathcal{C}')$. We will show that θ' is a solution of $\mathcal{C}'$. Let $(T' \Vdash u') \in \mathcal{C}'$ be the j^{th} constraint of $\mathcal{C}'$. We have that $T' = T_0 \cup \{v_{i_1}, \ldots, v_{i_{j-1}}\}$ and $u' = u_{i_j}$. By construction of $\mathcal{C}'$ and thanks to the fact that θ is a solution of $\mathcal{C}$, we know that the constraint $T_0, v_1, v_2, \ldots, v_{i_j-1} \Vdash u_{i_j}$ is a constraint of $\mathcal{C}$ and also that

$$T_0, v_1\theta, v_2\theta, \ldots, v_{i_j-1}\theta \vdash u_{i_j}\theta$$

Let $\mathsf{Names} = \{\mathrm{img}(\sigma_{r,s}) \cap \mathcal{N} \mid (r,s) \in \mathsf{sc} \text{ and } r > k_1\}$. Intuitively, Names are the fresh names generated during the execution of Π_2. Hence, we have that $\mathsf{Names} \cap \mathrm{n}(T_0) = \emptyset$ and $\mathsf{Names} \cap \mathsf{KC} = \emptyset$.

In the remainder of this proof, we will show that

1. $\overline{T_0}, \overline{v_1\theta}, \overline{v_2\theta}, \ldots, \overline{v_{i_j-1}\theta} \vdash \overline{u_{i_j}\theta}$, and then
2. $T_0 \cup \{v_{i_1}\theta, \ldots, v_{i_j}\theta\} \vdash u_{i_j}\theta$.

From this, we easily obtain that $T_0 \cup \{v_{i_1}\theta', \ldots, v_{i_j}\theta'\} \vdash u_{i_j}\theta'$ since $\theta' = \theta|_{vars(\mathcal{C}')}$ and $vars(\{v_{i_1}, \ldots, v_{i_j}, u_{i_j}\}) \subseteq vars(\mathcal{C}')$. This allows us conclude that $T'\theta' \vdash u'\theta'$ for any $(T' \Vdash u') \in \mathcal{C}'$, and thus θ' is a solution of $\mathcal{C}'$.

1. $\overline{T_0}, \overline{v_1\theta}, \overline{v_2\theta}, \ldots, \overline{v_{i_j-1}\theta} \vdash \overline{u_{i_j}\theta}$.

   By hypothesis, we have that $plaintext(\Pi_2) \cap \mathsf{KC} = \emptyset$, from this, we easily deduce that $plaintext(T_\beta) \cap \mathsf{KC} = \emptyset$. Thanks to Proposition 3, we have that $plaintext(x\theta) \cap \mathsf{KC} = \emptyset$ for every $x \in vars(T_\beta)$. Hence, we easily deduce that $plaintext(T_\beta\theta) \cap \mathsf{KC} = \emptyset$. Thanks to Proposition 3, we have also that $\mathrm{n}(T_\beta\theta) \subseteq \mathrm{n}(T_\beta) \cup \{\mathsf{init}\}$. Hence, we have that $\mathrm{n}(T_\beta\theta) \subseteq \mathsf{Names} \cup T_0 \cup \mathsf{KC}$. Putting all together, we obtain that $plaintext(T_\beta\theta) \subseteq T_0 \cup \mathsf{Names}$. Now, thanks to Lemma 8, we obtain that

   $$\overline{T_0}, \overline{v_1\theta}, \overline{v_2\theta}, \ldots, \overline{v_{i_j-1}\theta} \vdash \overline{u_{i_j}\theta}$$

2. $T_0 \cup \{v_{i_1}\theta, \ldots, v_{i_j}\theta\} \vdash u_{i_j}\theta$.

   We have that $\overline{T_0} = T_0$. Moreover, thanks to Lemma 7, we deduce that $T_0 \vdash \overline{v_i\theta}$ for every i such that $i \notin i_1 \cdots i_n$. To conclude, it remains to show that

- $\overline{v_i\theta} = v_i\theta$ and $\overline{u_i\theta} = u_i\theta$ for every $i \in \{i_1, \ldots, i_n\}$, and
- $\overline{m'\theta} = m'\theta$

In other words, we have to show that $\overline{w\theta} = w$ for any $w \in T_\alpha$. This fact is trivially true as soon as $w\theta$ is an α-tagged term such that $\mathrm{n}(w\theta) \cap \mathsf{Names} = \emptyset$. To conclude, it remains to show that $\mathrm{n}(w\theta) \cap \mathsf{Names} = \emptyset$ for any $w \in T_\alpha$.

Let $w \in T_\alpha$, we have that $\mathrm{n}(w\theta) \subseteq \mathrm{n}(T_\alpha\theta)$. Moreover, thanks to $(\star)$, we have that $\mathrm{n}(T_\alpha\theta) \subseteq \mathrm{n}(T_\alpha) \cup \{\mathsf{init}\}$. Hence, we deduce that $\mathrm{n}(w\theta) \subseteq \mathrm{n}(T_\alpha) \cup \{\mathsf{init}\}$. Thus, $\mathrm{n}(w\theta) \cap \mathsf{Names} = \emptyset$.

Hence, we easily deduce that $T_0 \cup \{v_{i_1}\theta, \ldots, v_{i_j}\theta\} \vdash u_{i_j}\theta$ by relying on the fact that $\overline{T_0}$, $\overline{v_1\theta}$, $\overline{v_2\theta}, \ldots, \overline{v_{i_j-1}\theta} \vdash \overline{u_{i_j}\theta}$. □

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399